

EEEEEEEEEE	XX	XX	AAAAAA	MM	MM	PPPPPPPP	LL	EEEEEEEEEE	SSSSSSSS
EEEEEEEEEE	XX	XX	AAAAAA	MM	MM	PPPPPPPP	LL	EEEEEEEEEE	SSSSSSSS
EEEEEEEEEE	XX	XX	AAAAAA	MM	MM	PPPPPPPP	LL	EEEEEEEEEE	SSSSSSSS
EE	XX	XX	AA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AA	AAAA	MM	PP	LL	EE	SS
EEEEEEEEEE	XX	XX	AA	AAAA	MM	PPPPPPPP	LL	EEEEEEEEEE	SSSSSSSS
EEEEEEEEEE	XX	XX	AA	AAAA	MM	PPPPPPPP	LL	EEEEEEEEEE	SSSSSSSS
EEEEEEEEEE	XX	XX	AA	AAAA	MM	PPPPPPPP	LL	EEEEEEEEEE	SSSSSSSS
EE	XX	XX	AAAA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AAAA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AAAA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AAAA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AAAA	AAAA	MM	PP	LL	EE	SS
EE	XX	XX	AAAA	AAAA	MM	PP	LL	EE	SS
EEEEEEEEEE	XX	XX	AAAA	AAAA	MM	PP	LLLLLLLLLL	EEEEEEEEEE	SSSSSSSS
EEEEEEEEEE	XX	XX	AAAA	AAAA	MM	PP	LLLLLLLLLL	EEEEEEEEEE	SSSSSSSS
EEEEEEEEEE	XX	XX	AAAA	AAAA	MM	PP	LLLLLLLLLL	EEEEEEEEEE	SSSSSSSS

```
LL      BBBB BBBB      RRRRRRRR      MM      MM      AAAAAA      CCCCCCCC
LL      BBBB BBBB      RRRRRRRR      MM      MM      AAAAAA      CCCCCCCC
LL      BB      BB      RR      RR      MMMM      MMMM      AA      AA      CC
LL      BB      BB      RR      RR      MMMM      MMMM      AA      AA      CC
LL      BB      BB      RR      RR      MM      MM      AA      AA      CC
LL      BBBB BBBB      RRRRRRRR      MM      MM      AA      AA      CC
LL      BBBB BBBB      RRRRRRRR      MM      MM      AA      AA      CC
LL      BB      BB      RR      RR      MM      MM      AAAAAAAAAA      CC
LL      BB      BB      RR      RR      MM      MM      AAAAAAAAAA      CC
LL      BB      BB      RR      RR      MM      MM      AA      AA      CC
LL      BB      BB      RR      RR      MM      MM      AA      AA      CC
LLLLLLLLLLLL BBBB BBBB      RR      RR      MM      MM      AA      AA      CCCCCCCC
LLLLLLLLLLLL BBBB BBBB      RR      RR      MM      MM      AA      AA      CCCCCCCC
```

```
MM      MM      AAAAAA      RRRRRRRR
MM      MM      AAAAAA      RRRRRRRR
MMMM      MMMM      AA      AA      RR      RR
MMMM      MMMM      AA      AA      RR      RR
MM      MM      MM      AA      AA      RR      RR
MM      MM      MM      AA      AA      RR      RR
MM      MM      AA      AA      RRRRRRRR
MM      MM      AAAAAAAAAA      RR      RR
MM      MM      AAAAAAAAAA      RR      RR
MM      MM      AA      AA      RR      RR
MM      MM      AA      AA      RR      RR
MM      MM      AA      AA      RR      RR
MM      MM      AA      AA      RR      RR
```

.TITLE demo_mac
 .IDENT 'V04=000'

```
*****
*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
*  ALL RIGHTS RESERVED.
*
```

```
*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
*  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
*  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
*  TRANSFERRED.
*
```

```
*  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
*  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
*  CORPORATION.
*
```

```
*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
*  *****
```

Macros

```
$credef      : Define create options array offsets
$dscredef    : Define string descriptor offsets
$lbrdef      : Define librarian parameters
$lbrctltbl   : Define library control table offsets
$namdef      : Define NAM block offset
```

```
Set up FORTRAN COMMON block to allow FORTRAN main program to
access librarian data
```

```
.PSECT lbrdata, PIC, OVR, REL, GBL, SHR, NOEXE, RD, WRT, LONG
```

```
.long lbr$c_read      : func_read
.long lbr$c_create    : func_create
.long lbr$c_update    : func_update
.long lbr$c_typ_txt    : type_text
.long lbr$c_typ_hlp    : type_help
.long rms$eof          : rmseof
.long dsc$e_class_d    : class dynamic
                        : offsets into create options array
                        : values are divided by 4 to convert byte
                        : offsets into longword offsets
.long cre$l_type/4      : type of library
.long cre$l_keylen/4    : max key length
.long cre$l_alloc/4     : initial library disk allocation
.long cre$l_idxmax/4    : number of indices
.long cre$l_uhdmax/4    : size of additional module header data
.long cre$l_entall/4    : number of index entries to preallocate
```


.SBTTL nam_init - Initialize RMS NAM block

++

Initialize array to be an RMS NAM block

Calling sequence:

call nam_init (nam_array, result_desc)

Inputs:

nam_array Address of array of ? bytes to be initialized
 as a NAM block

result_desc Address of string descriptor for resultant name
 string.

Outputs:

The nam_array is initialized as a NAM block, with the expanded
and resultant name strings pointing to the string described by
result_desc.

Routine value:

Always success

--

.PSECT \$code\$, PIC, REL, SHR, EXE, RD, NOWRT

.ENTRY nam_init, ^M<R2, R3, R4, R5, R6>

```

movl    4(AP), r6           ; Get address of NAM block
movc5   #0, (SP), #0, #nam$c_bln, (r6) ; Zero the NAM block
movl    8(AP), R0           ; Get address of resultant name string descriptor
$NAM_STORE NAM = R6, -     ; Initialize the NAM fields
        BLN = #nam$c_bln, - ; block length
        BID = #nam$c_bid, - ; block id
        RSS = dsc$w_length(R0), - ; resultant name string size
        ESS = dsc$w_length(R0), - ; expanded name string size
        RSA = @dsc$a_pointer(R0), - ; resultant name string address
        ESA = @dsc$a_pointer(R0), - ; expanded name string address
movl    #1, r0              ; Return with success
ret
.END

```


0157 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY